DOCUMENT RESUME

ED 178 054 IR 007 796

AUTHOR Muiznieks, Viktors
TITLE A General Introduction to Microcomputers.
INSTITUTION Illinois Univ., Urbana. Dept. of Secondary Education.
PUB DATE Nov 78
NOTE 19p.; The Illinois Series on Educational Applications of Computers, No. 26; For related documents, see IR 007 782, IR 007 845, and ED 142 199

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS Computer Programs; *Computers; *Computer Science; Computer Storage Devices; Mathematical Logic; *Numbers; Technical Education
IDENTIFIERS *Microcomputers

ABSTRACT

This basic introduction to microcomputers provides the neophyte with the terminology, definitions, and concepts that explain the microcomputer and computing technology in general. Mathematical operations with binary numbers, computer storage, controlling logic, and the concepts of stack and interrupt are explained. (RAO)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED178054

U.S. DEPARTMENT OF HEALTH
EDUCATION & WELFARE
NATIONAL INSTITUTE OF
EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY

Number 26

The Illinois Series

on

Educational Applications of Computers

A General Introduction to Microcomputers

Viktors Muiznieks

November 1978

Department of Secondary Education
College of Education
University of Illinois
Urbana, Illinois

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY

Bruce Hicks

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

IR006796

© Bruce Hicks (1978)

Preface

A draft of this report has been used for about a year by staff and students in our department. Many small improvements based on this experience have been incorporated in the present ISEAC Report by Shirley Moore and Kiyoko Miyazaki.

Bruce Hicks, Editor

ISEAC Number 26

TABLE OF CONTENTS

		Page
1.	What is a Microcomputer?	1
2.	What are the Basic Concepts?	3
3.	How are Bits Used in Microcomputers?	6
4.	Computer Memories	10
5.	How is the Computer Controlled?	12
6.	Advanced Concepts	14

ISEAC Number 26

A General Introduction to Microcomputers

Viktors Muiznieks

1. What is a Microcomputer?

Because of the widespread use and increasing complexity of microcomputer applications, no clearcut definition of the term MICROCOMPUTER really exists. In simplest terms, a microcomputer is a logic device. It is built up from various components including CHIPS, PRINTED CIRCUT CARDS, MEMORY and various PERIPHERAL DEVICES. The size and power of a microcomputer often depend upon how much memory the system has available, as well as the variety of peripheral devices which are attached to the system. The ways in which the microcomputer is to be used determine the specific kinds of devices which are or should be attached to it.

Microcomputers are based upon CHIP technology. All of the logic capabilities, the control, and the memory modules used in microcomputers are constructed from chips. The chips themselves are usually no larger than ½ inch x 1½ inches in size. They contain many small pieces of metal called transistors. Because transistors are much smaller than the vacuum tubes used in older computers, the term "micro" has been applied to computers which are built using them. Hence, what really has "shrunk" in microcomputers is the size of the components that handle the electronic signals going through the computer; other components, switches, consoles, and devices capable of accepting data and returning results to humans (the PERIPHERAL DEVICES) are still necessary and must be large enough to touch and see. Thus, the microcomputer system (the part that does the computing) may not appear to be as small as it actually is.

The term MICROPROCESSOR is often used to refer to a particular kind of chip, namely the one which contains the "heart" of the microcomputer system itself --

the central processing unit (CPU). All of the computing, arithmetic, and instruction execution take place within the CPU.

The MEMORY is also built out of specific kinds of chips. These chips allow a certain amount of information to be stored in them and also permit the CPU to retrieve the information that is stored.

In order to organize and connect all of the various kinds of chips together, PRINTED CIRCUIT CARDS are used. These are cards which have been designed to hold the various kinds of chips. The chips are soldered onto the cards in predefined places. Once the chips are in place, the connections (analogous to wires in older computers) which have been "imprinted" into the card serve to tie the chips together (complete a circuit between them) so that they can communicate properly. Often, it is also necessary to solder other standard circuit componenets, such as resistors, capaciters, and diodes frequently must also be soldered into the printed circuit card; however, all of these components will not be discussed here because they do not directly relate to this discussion.

After the printed circuit cards are built, they can be "plugged into" a central board which allows communication to occur between different cards. As soon as enough chips have been connected together (at least a microprocessor chip and some memory chips), a microcomputer system exists (at least in theory).

Traditionally, computers have been compared with respect to instruction sets, addressing modes, and execution speeds. These factors are of secondary importance to microcomputers, where the primary considerations are the ways in which the logic is distributed on chips and the prices of the various logic devices used.

In general, the cost of a chip is a function of its physical size, not of how much logic is implemented on it. A chip that has a number of GATES

(elementary building blocks defined in the next section) on it is called an INTEGRATED CIRCUIT (IC). If there are about 100-1000 gates on a chip, the logic on that chip is said to have medium scale integration (MSI). If the chip has more than 1000 gates on it, the logic on that chip is said to possess large scale integration (LSI).

2. What are the Basic Concepts?

As with all digital computers, the basic building blocks of the system center around devices that can be in one of two states -- either current is or is not flowing in a wire, something is or is not happening at a specific point in time, etc. This type of situation lends itself to being modeled in a BINARY way, where 0's and 1's can be used to indicate the states of the particular devices. For example, consider a switch -- the switch is either open —o/o— (corresponding to a 0) or closed —o-o— (corresponding to a 1). Each 0 or 1 is called a binary digit (abbreviated BIT), and binary operations exist between binary numbers just as they do for decimal numbers. The only difference is that the operations are being performed in base 2 instead of base 10, so that only the digits 0 and 1 exist, instead of all the digits 0-9.

Hence, to find the equivalent decimal value of binary numbers you could use (and expand upon) a chart like the following:

Place value	$2^3=8$	$2^2=4$	$2^1=2$	$2^0=1$	decimal equivalent of binary 2^n
	0	0	0	0	=0
	0	0	0	1	=1
	0	0	1	0	=2
	.	.	.	.	
	.	.	.	.	
	.	.	.	.	
	1	1	0	1	=13
	1	1	1	0	=14
	1	1	1	1	=15

Addition of binary numbers is defined in the following way:

0+0=0 1+0=1 0+1=1 1+1=0 (with a carry of 1). Example:

```
  11001010
+ 1100101
=100101111
```

Subtraction of binary numbers obeys the following rules:

0-0=0 1-0=1 1-1=0 0-1=1 (with a borrow). Example:

```
  10001
- 1011
=00110
```

Multiplication of binary numbers obeys the following rules:

0x0=0 0x1=0 1x0=0 1x1=1. Example:

```
   1101
x  101
   1101
  0000
 1101
=1000001
```

Binary division can be done in the same way as long division of decimal numbers. Example:

```
        1011
   ------------
101 / 110111
      101
      ----
      0011
      0000
      ----
       111
       101
       ---
       0101
        101
        ---
          0
```

When dealing with binary numbers, other operations besides the normal addition, subtraction, multiplication, and division operations are possible. The most common of these are NOT, AND, and OR. The definitions of these operations are usually given in the form of a TRUTH TABLE, illustrated below.

NOT

A	NOT A
0	1
1	0

AND

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

OR

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

For example,

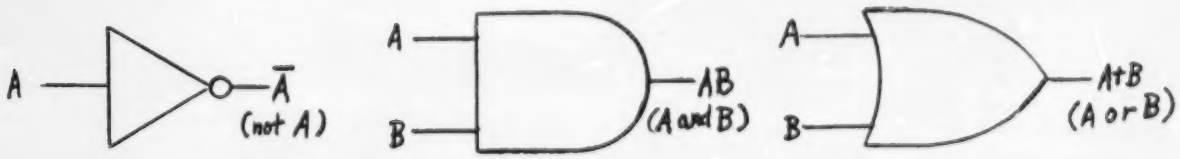
$$\text{NOT } \frac{10111001}{} = 01000110$$

$$\text{AND } \frac{10111001}{01100011} = 00100001$$

$$\text{OR } \frac{10111001}{01100011} = 11111011$$

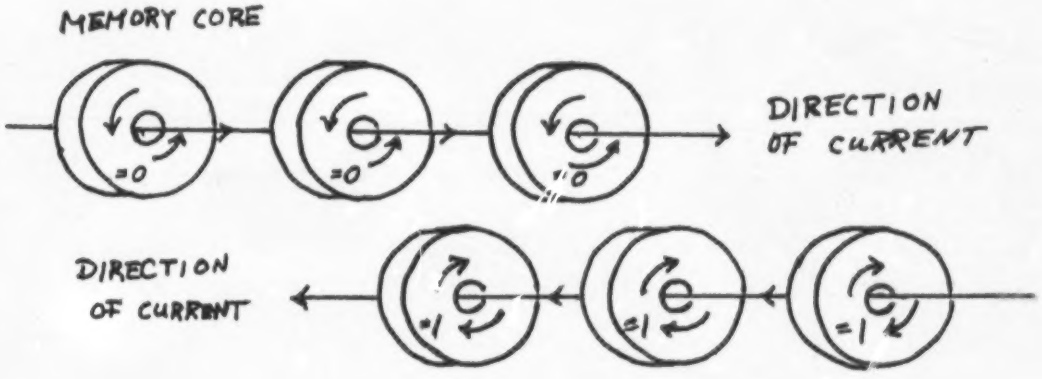
Often the word COMPLEMENT is used to mean "NOT each bit" in the number given.

Inside the computer, the above operations are accomplished inside GATES, one or two bits enter a gate, and a result bit is produced according to the rules outlined above. These gates are usually symbolized in the following way:

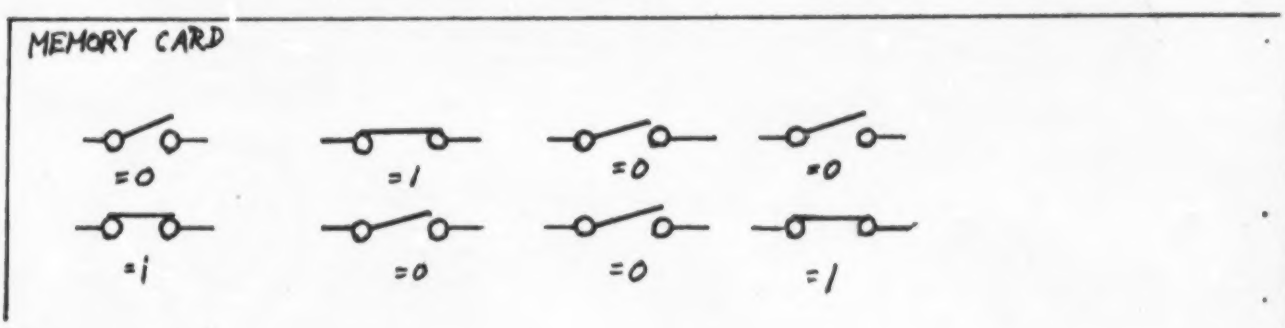


3. How are Bits Used in Microcomputers?

First of all, the bits which the microcomputer will use must be stored somewhere. In older computers, bits are commonly stored in "core" memories, where tiny ferrite (iron) "doughnuts" have a clockwise or counterclockwise magnetic field (the direction of the magnetic field corresponds to either a 0 or a 1).



Microcomputers have advanced to the point of being able to use semiconductor memories, in which each bit can be represented by a switch. Switches can be either open or closed (the status of the switch determines if the bit is a 0 or a 1).



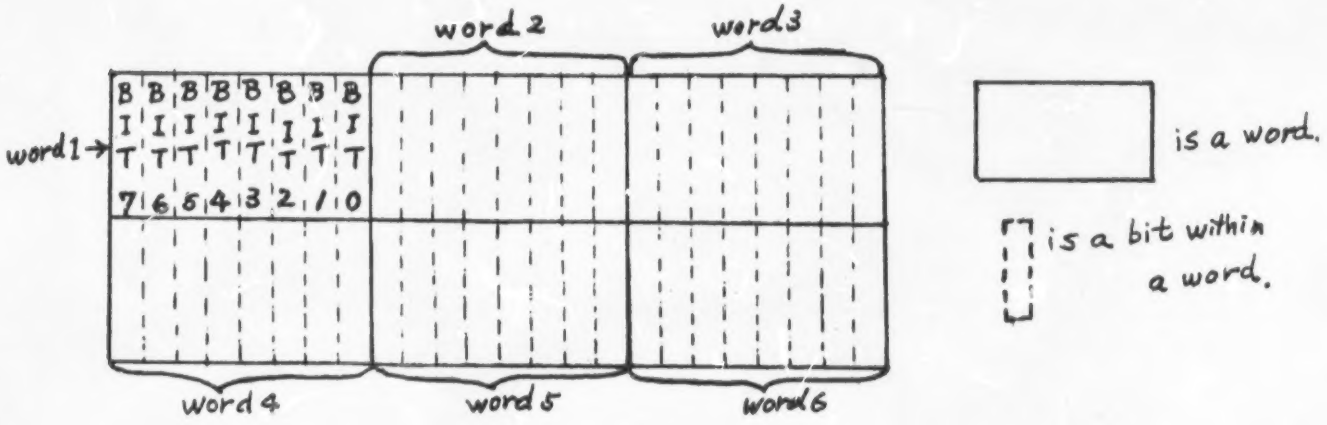
There is a significant difference between these two kinds of memories (places where bits can be stored). Magnetic core memories retain their sense of direction even when disconnected from the electrical power source required to run the main computing activities. Hence, core memories could conceivably be removed from one computer and attached to another and still contain the same information as on the original system. This kind of memory is called NON-VOLATILE, since the information stored there cannot be lost because of a removal of electric power.

Semiconducter memories, on the other hand, are susceptible to losing the information stored in them if power is turned off. Therefore, the information stored in the memory of a microcomputer cannot be easily transferred to another microcomputer. In fact, one cannot be sure what will be stored in the memory of a microcomputer if the power source is turned off momentarily and then turned back on again. (Hence, be careful turning off power!) The memories that lose their information when power sources are removed are called VOLATILE memories.

There is one other important distinction that is often drawn between different kinds of memories. This distinction results from the fact that while each bit that is stored in all memories can be uniquely accessed (i.e., read and interpreted as a 0 or a 1), not all memories can have the values of their bits changed after they are initially established. Memories in which the state of the bits cannot be changed are referred to as read-only memories (ROM); naturally, read-only memories are always NON-VOLATILE. Memories in which the status of bits can be altered as well as read are referred to as read-write or random access memories (RAM).

Because most computers are relatively complex, meaningful information usually requires more than 1 bit for its expression. It has become customary to

group bits into more useful units called WORDS. Whenever the memory is accessed, complete words of information, instead of single bits, are retrieved or stored. It has also become necessary to establish some means for ordering the bits within a word, and the most common convention is to treat the rightmost bit as bit 0 of the word (it is usually the least significant digit in any number represented by a computer word, i.e., it corresponds to the lowest power of 2 being used in the representation) and the leftmost bit as the most significant bit in the word. For example, if a computer has a word length of 8 bits, then the memory might be organized as follows:



Some common word sizes are given in the table below so that you may acquire a feeling for the normal word sizes available in various kinds of computers.

Word size in bits	Microcomputers	Minicomputers	Large Computers
4	many	none	none
6	none	a few old models	none
8	most common	a few	none
12	a few	a few	none
16	a few	most common	a few
18	none	a few	a few
24	none	a few	a few
32	none	a few	most common
64	none	none	some of the largest ones

Minicomputers are computers whose size and power fall somewhere between the extremes represented by microcomputers and very large computer systems.

Another way of referring to groups of bits is by BYTES. A byte usually consists of 8 bits (although 6- and 7-bit byte machines do exist). Bytes are the most common way of representing symbols in the computer. Computer words are constructed from bytes. Some computer words contain several bytes of information, whereas words on other computers are the same length as bytes. (On such machines, bytes and words can be thought of as being synonymous.)

A computer would not be very useful if all the data to be fed into it had to be entered as binary digits. This is where bytes become useful. The symbols we would most like to be able to use to communicate with a computer are the 26 letters of the alphabet (in both upper and lower cases), the digits 0-9, and about 25 special symbols like %, $, @, etc. There are about 100 different symbols we would like the computer to be able to recognize. In order to uniquely identify 100 symbols in a binary number system, at least 7 binary digits must be used (since 2^7 is 128 and 2^6 is only 64). Therefore, it makes sense to group 7 bits together to form a pattern which corresponds to each of those symbols, with each pattern of 0's and 1's corresponding to a different symbol. This is what computer designers have done; hence, when you instruct a computer to do something, the instructions you give it are translated into the bit patterns the computer understands. This translation involves forming bytes that are equivalent to the symbols you used. The reason bytes are usually 8 bits long instead of 7 is that the extra bit is used to provide a means of checking errors in data transmission.

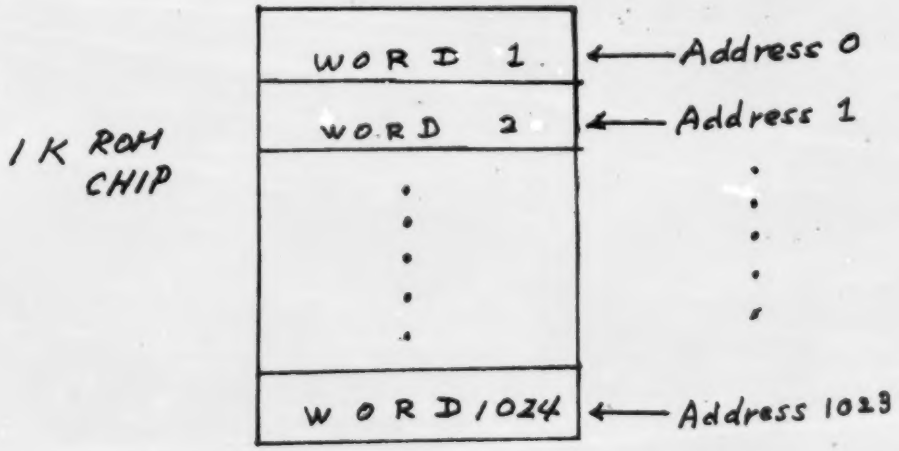
Error checking is usually done by looking at the number of 1's in the 7 bits of the byte representing the symbol or data -- if there are an even number

of 1's in the 7 bits, then the 8th bit is set to a 0; if there are an odd number of 1's, then the 8th bit is set to a 1. This procedure is called determining the parity (whether there are an even number of 1's) of the 7 bits. An error condition is signalled whenever the parity condition of a byte being used is not satisfied.

When a computer has a byte size of less than 8 bits, it means either that no error checking (using a parity scheme) is done or that a limited number of symbols must be used. The microcomputers that use 4-bit words form a byte out of two words. The computers that have word sizes greater than 8 bits can store more than one byte of information in a word, but those machines for which the word size is not a multiple of 8 either waste a few bits in the representation or store parts of bytes in different words in a complex way.

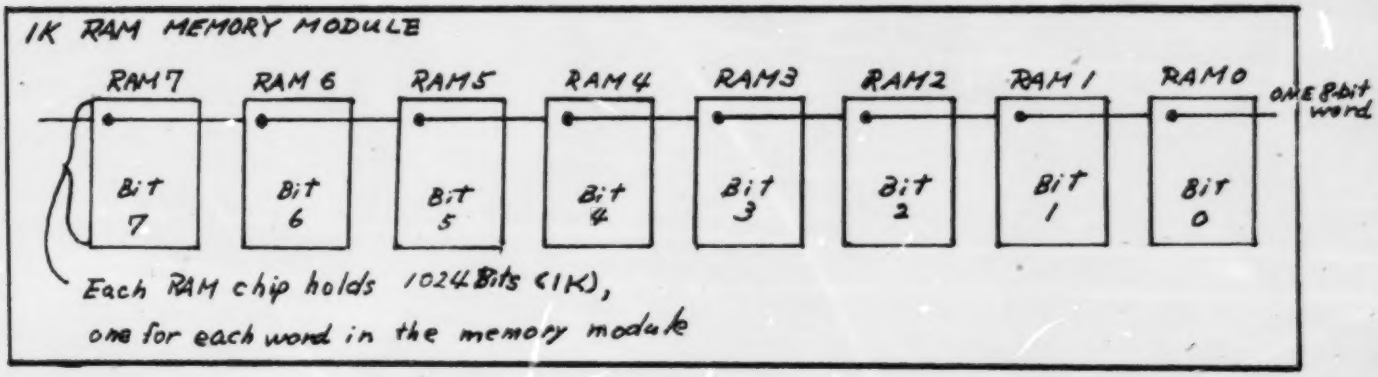
4. Computer Memories

Information entered into the computer is stored in the computer memory. The computer memory is divided into individual areas capable of holding one computer word, and each word in the memory is assigned a unique ADDRESS. Addresses start with 0 and increase by 1 up to the maximum number allowed; the biggest address will vary with the size of the memory. The address of a word corresponds to its location in memory as illustrated below.



In microcomputer ROM memories, it is common to have 1024 (2^{10}=1K) words on a single chip. Most microcomputers have an 8-bit word, consequently 8192 binary digits can be stored on 1 ROM chip. The way in which the words are stored may be thought of in a fashion represented in the above diagram, where each word is stored sequentially.

Since RAM memories require more logic to allow for the alteration of their contents, RAM memories are often implemented on more than one chip. A common way of doing this is illustrated by the diagram below, where eight chips are used. Each bit in a word (byte) is represented in a separate chip, and all eight chips serve together as one memory module.



Other methods for implementing RAM memories are possible. In general, the cost (in terms of cents per bit) increases as the number of chips used to implement a single word decreases. Hence, the preceding diagram illustrates the cheapest way of storing 8-bit words; the most expensive way to implement an 8-bit RAM memory would be to use a single chip.

There are basically two types of RAM memories -- dynamic and static. A bit stored in dynamic RAM must be refreshed every few seconds. Dynamic RAM is cheaper to buy, but refreshing logic must also be bought and integrated into the

computer logic, so that the information stored in a bit is not lost within a few milliseconds after it is put there. Static RAM is more expensive, but once data is put into a bit, it remains there until it is changed or the power source is removed.

The information that is stored in any computer word can be one of the following four types:*

1) A pure binary number,
2) A piece of numeric data that is part of a multiple word data item,
3) A data code for one of the symbols discussed on pp. 8-9,
or 4) An instruction code, which tells the computer what operations to perform.

One cannot tell what kind of information is stored in a memory word by merely looking at it. The way a memory word is interpreted depends upon how the computer is told to interpret it. This complex subject will not be treated at this time.

5. How is the Computer Controlled?

The logic of the microcomputer (which makes possible its various operations) is also implemented on one or more chips. These chips have PINS attached to them out of which various kinds of signals can flow and through which signals from other parts of the system can be received. The controlling logic of the system is commonly referred to as the central processing unit (CPU), which carries out several functions.

Before the CPU can perform any sort of operation on data, it must have a place to store this data. Hence, storage locations within the CPU similar to those found in the memory must be provided. These storage locations are called REGISTERS to distinguish them from normal memory locations. All microcomputers

*For further information see An Introduction to Microcomputer by Adam Osborne and Associates, Incorporated, 1975. pp. 3-10 - 3-20.

have a small number of registers available which can be used to store data items and intermediate results.

Most microcomputers have registers which are designated for specific purposes. For example, an ACCUMULATOR is a register employed to store both the operands used in a numerical operation and the operands that result from the operation. Since information can be stored in the accumulator, the number of memory references is decreased. The computer can work faster when it does not have to reference the memory many times for each operation. (Memory references take a lot of time, in terms of internal CPU speeds.)

In order to store information into or retrieve data from a memory location, the CPU must know the address at which the data is to be stored or from which it is to be retrieved. This address is kept in a special register known as the PROGRAM COUNTER.

Similarly, the type of operation which the computer is to perform must also be known; consequently, the code corresponding to the instruction that is to be performed is stored in a register known as the INSTRUCTION REGISTER.

The problem of "How does the computer know where to get the next instruction?" is usually resolved in the following way. There is a special way in which the computer knows where to get the very first instruction; after that, the computer expects successive instructions to come from successive locations in memory. When an instruction is executed, the program counter is incremented by 1, and the computer can get the next instruction. There are special means built into the system by which this successive flow can be modified, but they need not be discussed at this time.

Some microcomputers have other special registers with important functions but which need not be discussed here. Aside from the special purpose registers

just described, most microcomputers have from four to eight programmer-controlled registers. These are registers which a programmer can use in programs to keep track of information or to perform certain operations (with a resulting increased power and speed of the CPU).

Every CPU has an associated ARITHMETIC and LOGIC UNIT (ALU) which can actually perform the arithmetic and Boolean (AND, OR, NOT) operations, complement data words, and shift the bits of a data word either to the right or to the left. More complex operations can be constructed from combinations of these simple operations. In order to sequence properly the logic elements of the ALU to implement a given operation, the CPU must also contain some kind of CONTROL UNIT which is capable of interpreting the actions specified in the instruction register.

6. Advanced Concepts

There are two other important concepts which belong to the microcomputer world, neither of which is new in computer technology. The first of these is the concept of a STACK. A stack is an area of memory set aside for the storage of data and addresses. The stack is addressed by a special register known as the STACK POINTER which points to the location of the last item placed onto the stack. Usually only the following two operations may be performed with a stack: 1) writing something onto the top of the stack (referred to as a PUSH operation); or 2) reading the top item of the stack (referred to as a POP operation). Hence, only the last item placed on the stack can be accessed; in order to get information from the middle of the stack, all of the preceeding information must also be read. Stacks have many applications in the microcomputer world. One important area where they are indispensible is that of subroutine calls.

When a subroutine call is made, all of the values stored in the registers are placed into the stack, and the address of the beginning of the subroutine

is placed into the program counter. The instructions in the subroutine are executed until a statement indicates that a return to the main program should occur. At this point, the register values are retrieved from the stack, and the program can resume execution where it left off.

The second important concept is that of an INTERRUPT. An interrupt is simply a request from some device or program for the computer to stop what it is doing and do something else. It is an unscheduled event that requires immediate computer attention. An interrupt is treated in a manner similar to that just described for a subroutine call -- the contents of the registers are saved on the stack, and then control is transferred to the place specified by the interrupting device or program. When a "return-from-interrupt" instruction is found, the main program is allowed to continue as soon as the contents of the registers have been restored from the stack.

Of course, many more things besides those outlined in this paper go on inside a microcomputer; however, most microcomputers have the features and characteristics described in this brief introduction.